퍼지 및 지능시스템학회 논문지 2006, Vol. 16, No. 4, pp. 485-492

휴머노이드 로봇의 분산 제어를 위한 네트웍 구현

Network Realization for a Distributed Control of a Humanoid Robot

이보희*, 공정식**, 김진걸***
Bo-Hee Lee, Jung-Shik Kong, and Jin-Geol Kim

* 세명대학교 전기공학과
** 인하대학교 자동화공학과
*** 인하대학교 전자전기공학부

요 약

본 논문은 휴머노이드 로봇 ISHURO의 분산 제어를 위한 네트웍 구현에 대해 다루고 있다. 일반적으로 휴머노이드형 로봇은 기구학적으로 유연한 동작을 위해 다수의 자유도가 필요하다. 이를 구현하기 위해서는 중앙에서 일괄적으로 처리 하는 것 보다 간결 하면서도 유연성을 줄 수 있는 분산 처리 방법이 선호되고 있다. 분산 처리를 위한 제어기를 구성할 때는 로봇의 모터를 독립적으로 제어하기 위한 제어기가 별도로 필요하며 모듈 간에는 정해진 시간 내에 데이터를 교환할 수 있는 통신 기법이 필요하다. ISHURO의 각 관절은 자체 내에 독립된 DSP를 내장하고 있으며 CAN 네트웍을 이용하여 모듈간의 통신을 하여 구동기를 제어하거나 센서의 값을 모니터링 할 수 있게 되어 있다. 본 논문에서는 이를 위한 통신 구조를 제안하고 필요한 전송 메시지를 정의하고, 전송시간을 분석하여 로봇 분산 제어기 구조에 적절한 전송 프로토콜을 제시하였다. 모든 과정은 Matlab을 이용하여 컴퓨터모의실험을 수행하였고 실제 휴머노이드 로봇에 적용하여 보행실험을 통해 검증 하였다.

키워드 : CAN 네트웍, 휴머노이드 로봇, 분산제어, DSP 기반 프로세서

Abstract

This paper deals with implementation of network for distributed control system of a humanoid robot ISHURO(Inha Semyung Humanoid Robot). A humanoid robot needs much degree of freedom structurally and much data for having flexible movement. To realize such a humanoid robot, distributed control method is preferred to the centralized one since it gives a compactness, modularity and flexibility for the controllers. For organizing distributed control system of a humanoid robot, a control processor on a board is needed to individually control the joint motor and communication technology between the processors is required to transmit its information within control time. The processor is DSP-based processor and includes CAN network on a chip. It shares the computational load such as monitoring the sensor information and controlling the actuator between each of modules. In this paper, the communication architecture is suggested and its message protocol are discussed including message structure, time consumption for transmission, and controller structure at the view of distributed control for a humanoid robot. All of the sequence are simulated with Matlab and then verified with real walking experiment by ISHURO.

Key words : CAN network, Humanoid robot, Distributed Con trol, DSP-based processor

1. 서 론

최근 다양한 지능형 로봇이 개발되고 있다. 지능형 로봇은 개인용 로봇, 전문가용 로봇으로 분류할 수 있고 이동성을 가진 로봇은 바퀴 구동형 로봇과 다리를 갖는 로봇으로 구별되고, 다리를 가진 로봇은 다리의 개수에 따라 도약 형의 외발로봇, 인간과 유사한 이족보행로봇, 동물과 같은 구조의 사족보행로봇내지 육족보행로봇으로 구별된다. 그 중에서도 인간과의 대화, 접촉 또는 조작 등에 반응하고 인간의 생활환경에 적응할 수 있는 인간 형태의 로봇에 관한 연구가 활발히 진행 중이다. 휴머노이드 로봇은 인간과 유사한 구조를 가지므로 인간과의 협동 작업 및 인간이 수행하기 어렵고 위험한 작업을 대신할 수 있다[1-3].

이러한 휴머노이드 로봇이 인간의 생활공간에서 자율적인 보행을 수행하기 위해서는 외부환경을 인식하고 인식된 정보로부터 동작을 정의하고, 모든 자유도의 궤적을 생성하는 연산 모듈과 시스템을 구동시키기 위한 제어 모듈을 가지고 있어야 한다. 그리고 로봇의 이동성을 보장하기 위해서는 센서들과 연산 모듈이 로봇 내부에 탑재되거나 인식 장치는 로봇 내부에 탑재되고 연산 모듈은 로봇 외부에 두고 로봇과는 무선으로 연결되어야 한다. 로봇 내부에 센서 및 연산 모듈이 장착된 경우는 실시간 동작에서의 장점이 있지만 로봇 구조

접수일자 : 2006년 3월 23일
완료일자 : 2006년 8월 14일
감사의 글 : 본 연구는 한국과학재단 목적기초연구 (R01-2003-000-10364-0) 지원으로 수행되었음

가 복잡하게 된다. 반면에 외부에 연산 모듈을 두는 방식은 로봇 구조가 간단해지지만 빠른 무선 프로토콜이 요구되고 실시간 동작에 제한을 받게 된다[4].

로봇 내부에서 보행에 필요한 데이터를 생성하는 경우, 모든 자유도들에 대한 데이터를 생성할 수 있는 가볍고 빠른 연산 시스템이 필요하다[5, 6]. 휴머노이드 로봇은 구조적으로 많은 자유도를 필요로 하고 인간과 같은 유연한 보행을 하기 위해서는 각 자유도마다 많은 양의 데이터를 필요로 하고 로봇의 동작에 필요한 데이터들은 연산모듈에서 생성된 후 액추에이터를 가진 각각의 하위 모듈로 빠르고 정확하게 전송되어야 한다. 이러한 작업은 휴머노이드 로봇이 인간과 같은 작업성을 가지기 위한 필수 요건이다. 이러한 요건을 만족하려면 각각의 데이터에 대한 응답시간분석과 전송 시 에러 발생에 관한 연구가 선행되어야 하기 때문에 네트워크 프로토콜 및 전송 시간에 관한 연구가 활발히 진행되고 있다[7-11]. 이때 로봇의 보행 데이터의 전송 안정성을 확보하기 위해서는 CAN통신 특성에 따른 최악의 상황을 고려한 데이터의 전송 시간 계산이 필요하다[12, 13].

본 논문에서는 휴머노이드 로봇의 제어 신호와 보행 데이터 전송에 필요한 네트워크를 CAN프로토콜을 이용하여 설계하고 정의된 CAN메시지를 스케줄링 하여 버스 상태를 모의 실험을 통해 확인하고 로봇에 적용하는 것을 목적으로 한다. 이를 위해 본 논문에서의 II장에서는 ISHURO의 시스템 구성과 하드웨어 특성 및 시스템에 사용된 CAN하드웨어를 설명한다. III장에서는 CAN 프로토콜을 이용한 분산 제어 시스템을 설명하고 CAN 메시지 정의한다. IV 장에서는 모의 실험기를 이용하여 계획된 CAN 메시지의 전송 가능성 및 제어기의 확장성을 검증한다. V 장에서는 ISHURO시스템을 이용한 실험 결과를 설명한다. 마지막으로 VI장에서는 결론을 제시한다.

2. 시스템 구성

2.1 ISHURO의 하드웨어

그림1은 21개의 자유도를 가지는 휴머노이드 로봇 ISHURO의 기본 구성을 나타낸다. 각 다리에 6개의 자유도, 각 팔에 3개 자유도, 몸체에 한개 자유도, 머리에 2개 자유도를 가진다. 각각의 관절은 소형 직류 모터로 구동되고 모터 종단부 및 기구종단부에는 엔코더를 장착하여 회전각을 계측하게 되어 있으며 계측된 센서 값은 CAN통신을 이용하여 주 제어기에 전송되는 구조로 제작 되었다.

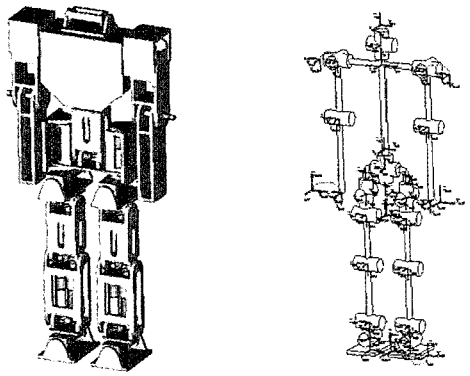

그림 1. ISHURO 시스템 좌표계
Fig. 1. Coordination of a humanoid robot

ISHURO의 분산 처리를 위해 두 가지의 통신 방식이 적용되었다. 첫 번째 방식은 로봇의 주 제어기와 로봇 외부 모의 실험기와의 통신을 위한 무선랜(802.11b)이고 두 번째는 각각의 지역 제어기와 주 제어기와의 통신을 위한 CAN이다. 로봇의 발바닥에는 반발력과 ZMP(Zero Moment Point)를 측정하기 위한 센서가 부착되었다. 표 1은 ISHURO의 제원을 나타낸다. 전체 높이는 825mm, 최대 폭 330mm, 최대 너비 135mm이며 전체 무게는 약 6.84kg으로 설계되었다.

표 1. ISHURO 사양.
Table 1. Hardware specification of ISHURO.

Height	825 mm	
Width	330 mm	
Foot	135 mm	
D.O.Fs	21	
Actuator	A-max26	Re-max24
Power	18V, 11W	18V, 11W
Gear Ratio	231:1	190:1
Angular Velocity	230deg/s	240deg/s

ISHURO의 구동부에는 Maxon A-max26, Re-max24 2종류의 모터가 사용되었고, 각각 231:1, 190:1의 기어 감속기가 장착되었다.

2.2 CAN 하드웨어

휴머노이드 로봇은 다축 구동 시스템이며 구동기로 사용되는 모터를 제어하려면, 서보 제어 기술과 계측 기술이 필요하다. 로봇의 관절은 서로 결합된 구조를 가지고 큰 토크를 얻기 위해 감속기가 사용되었고 이로 인해 발생하는 백래쉬 현상을 극복하기 위한 고급제어 기술이 필요하다. 그리고 구동축이 다수인 관계로 한 개의 고성능 CPU로 모든 구동축을 제어하는 방법보다는 여러 개의 CPU를 이용한 분산 제어 방법이 제어기의 구조도 간단해 지고 가격, 확장성에서 장점을 가진다.

통신방법에 있어서 병렬접속을 하면 속도 면에는 이점이 있으나 접속회로가 복잡해지고 단점이 있어 접속회로가 간단한 직렬접속 방법을 이용한다. 여러 가지 직렬 통신 방식 중에서도 CAN 프로토콜은 작은 데이터 전송에 안정성과 속도 면에 장점으로 인해 자동차, 로봇, 산업 자동화 등과 같은 다양한 분야에서 많이 사용되고 있다. 이러한 점은 ISHURO에서도 로봇 관절이나 전력에서 발생할 수 있는 노이즈에 강성을 가지고 데이터 신뢰성을 확보할 수 있으며 하드웨어와 독립되어 있으므로 하드웨어 성능에 따로 영향을 미치지 않는다. 이러한 이유로 CAN프로토콜을 이용하여 ISHURO의 통신 네트워크를 구현하였다.

그림 2는 ISHURO의 주 제어기이다. 주 제어기는 고성능 프로서인 Intel PXA255 Processor(400MHz)를 탑재하고, OS는 임베디드 리눅스를 사용하였다. 64Mbyte의 SDRAM과 32Mbyte의 Flash가 장착되어 있고 PCMCIA 타입의 무선랜을 내장하고 있어 외부 모의 실험기와 802.11b 무선통신이 가능하다. Infineon사의 8bits CAN 컨트롤러 81C91 탑재하여 로봇의 각각의 지역 제어기와의 CAN 기반의 통신을 수행한다.

그림 3은 ISHURO의 지역 제어기의 외관을 보여 주고 있다. 지역 제어기는 32비트-TMS320F2812 DSP를 탑재하고

있으며 자체 내에 플래시 메모리를 내장 하고 있으며 eCAN 모듈이 장착되어 있기 때문에 외부의 별다른 CAN접속부가 필요 없다. 분산 제어를 실현하기 위해서 지역 제어기는 하나의 모터만 제어하게 하였으며 모터에 필요한 PWM회로 및 엔코더 상 검출 회로도 자체 내에 내장 되어 있다. 본 연구에 사용된 제어기는 1개의 주 제어기와 각 관절을 구동하는 21개의 지역 제어기들로 구성되어 있다. 지역제어기와의 통신은 CAN 기반으로 되어 있으며 로봇의 자체의 무게 중심 및 자세를 보정하기 위한 센서 접속 모듈로도 사용할 수 있게 설계 되었다.

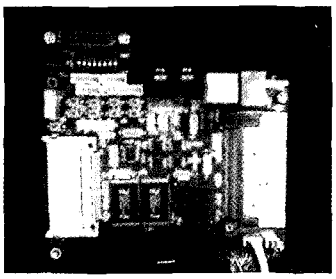

그림 2. 주 제어기
Fig. 2. Main controller of ISHURO

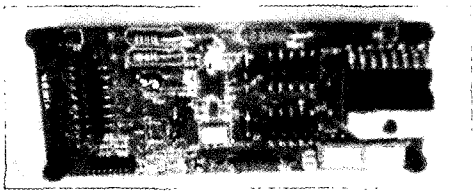

그림 3. 지역 제어기
Fig. 3. Local controller of ISHURO

3. CAN 기반의 분산 제어 시스템

3.1 CAN 프로토콜 개요

CAN은 자동차 내의 ECU(Electronic Control Unit)와 센서, 액츄에이터 간의 디지털 통신을 제공하기 위하여 1980년대 보쉬에서 개발되었으며, 1993년 ISO 표준으로 제정되었다. CAN은 다른 필드버스 프로토콜에 비하여 가격 대 성능비가 매우 우수할 뿐만 아니라, 대다수의 반도체 제조 회사에서 원 칩 형태의 마이크로 컨트롤러로 개발되어 가장 대중화되어 있는 프로토콜로 인정받고 있다. 이러한 장점으로 인하여, CAN은 자동차 분야뿐만 아니라, 공장 자동화와 빌딩 자동화, 로봇, 선박, 공작기계 등과 같은 다양한 분야에서의 데이터 교환을 위한 응용으로 확대되어 가고 있다[14, 15]. 그림4는 CAN 프로토콜의 간략화 된 동작 흐름을 나타내었다.

CAN프로토콜은 메시지의 내용에 따라 11비트 또는 29비트로 이루어진 식별자(identifier, ID)를 부여하여 모든 메시지를 구별하고 메시지의 우선권을 정하는 내용 기반 주소 지정 방식(content based addressing)을 사용한다. 그림 5에서 만약 어떤 스테이션이 메시지를 전송하기 시작하면, 나머지 스테이션 들은 수신된 데이터가 자신이 가진 ID와 관련이 있는 지를 비교한다. 만약, 자신이 가진 ID와 관련이 있으면 받아들이고, 관련이 없으면 무시한다. 또한, 둘 이상의 스테이션에서 거의 동시에 전송을 시도 했을 경우에는 충돌이 발생할 수 있다. 이러한 상황에서 CAN은 충돌된 메시지들의 ID를

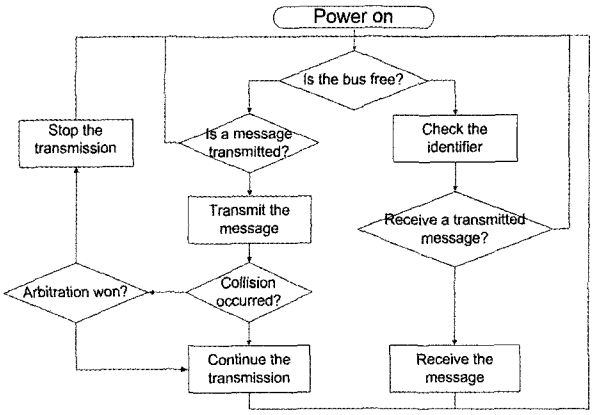

그림 4. CAN 프로토콜의 흐름도
Fig. 4. Flowchart of CAN protocol

한 비트씩 비교하여 가장 낮은 ID값을 가진 메시지(우선순위가 가장 높음)만 전송을 계속하고, 나머지 메시지는 즉시 전송을 중단하는 CSMA/NBA(Carrier Sense Multiple Access with Non-destructive Bitwise Arbitration)을 사용한다[16]. CAN을 이용한 데이터 전송 시 데이터 전송 시간을 계산하기 위한 수식은 식(1)과 같다[17].

$$R_i = J_i + q_i + C_i \tag{1}$$

R_i는 최악의 상황에서의 메시지 i의 전송 시간을 나타내고, C_i는 메시지 i가 물리적인 버스를 통해서 전송되는데 걸리는 시간이고, CAN 메시지는 47bits의 오버헤드 및 비트 스터핑(bit-stuffing)이 필요하다. 여기서 C_i는 식(2)와 같이 표현된다.

$$C_i = \left(\left\lfloor \frac{34 + 8s_i}{5} \right\rfloor + 47 + 8s_i \right) \tau_{bit} \tag{2}$$

식(2)에서 s_i는 메시지 i의 크기(byte)를 나타내고 τ_{bit}는 1bit 데이터가 전송되는 데 걸리는 시간을 의미 한다(1Mbps를 사용하는 버스에서는 1 μs). 식(3)은 최악의 상황에서 메시지가 전송 큐에서 소비하는 시간(q_i)을 나타낸다. hp(i)는 시스템 상의 메시지 i보다 우선순위가 높은 메시지의 집합이다.

$$q_i^{n+1} = B_i + \sum_{\forall m \in hp(i)} \left\lceil \frac{q_i^n + J_m + \tau_{bit}}{T_m} \right\rceil C_m + E_i(q_i + C_i) \tag{3}$$

식(4)는 메시지 i보다 우선순위가 낮은 메시지의 블록킹 시간 B_i를 나타낸다.

$$B_i = \max_{\forall k \in lp(i)} (C_k) \tag{4}$$

$lp(i)$는 시스템 상의 메시지 i보다 우선순위가 낮은 메시지의 집합이다. J_m은 메시지의 지터, T_m은 메시지의 주기이다. 식 (5)는 메시지가 큐에서 소비하는 시간 중에서의 에러 복원 오버헤드 함수를 나타낸다.

$$E_i(t)=\left(n_{err}+\left\lceil\frac{t}{T_{err}}\right\rceil-1\right)\left(29+\max_{\forall k\in hp(i)\cup\{i\}}(C_k)\right) \quad (5)$$

식(5)에서 n_{err}는 시간 T_{err} 동안 발생하는 에러 횟수이다. 여기서 매번 에러가 발생할 때는 다시 전송을 위해 29bit로 증가하게 만든다.

3.2. CAN 메시지 정의

휴머노이드 로봇에 CAN 프로토콜을 적용하기 위해서는 로봇동작에 필요한 데이터를 정의하고 정의된 데이터를 CAN 메시지로 만드는 작업이 필요하며, 필요한 데이터 정의하기 위해서 로봇의 상태를 보행 전 초기화 상태와 보행 상태로 구분해야 한다. 보행 전 초기화 상태의 메시지와 보행 상태의 메시지는 표2와 같이 표현된다. 보행 전 초기화 상태는 ISHURO가 보행을 하기 위한 초기 자세를 설정해 주는 과정이다. 먼저 사용자 명령에 의해 모든 지역 제어기의 상태를 확인하게 된다. 이 때 주 제어기에서 지역 제어기로 Module_State 메시지를 전송하고 이 메시지를 수신한 모든 지역 제어기들은 주 제어기로 확인 신호를 보내어 자신의 상태를 알리게 된다. 다음에는 각 구동부의 출력 축 엔코더의 I-상을 이용하여 로봇이 똑바로 선 상태를 만들어 준다. 이 때 사용하는 메시지가 Output_I_State 이다. ISHURO가 똑바로 선 상태에서 초기 보행을 위한 무릎을 구부린 초기 자세를 취하기 위해 주 제어기에서 각 지역 제어기로 초기 위치 데이터를 Walking_Position 메시지를 이용하여 전송하여 보행 초기 상태를 유지하게 된다. 초기화 마지막 단계로 출력 엔코더 한 펄스 움직임으로 초기 위치를 보정하게 된다.

표 2. ISHURO의 메시지 정의
Table 2. Message definitions of ISHURO

초기상태	
Message Name	상태
Module_State	모듈 상태 점검 메시지
Output_I_State	출력축 엔코더 I-상 검출 메시지
Walking_Position	보행 초기 위치 이동 메시지
Jog_Move	1 펄스 이동 메시지
보행 상태	
Message Name	상태
Global_Sync	모터 구동 메시지
Angle_Data	보행 궤적 데이터 메시지
Encoder_Data	엔코더 데이터 메시지
Sensor_Data	(FSR, Gyro) 센서 데이터 메시지

보행 상태에서는 모든 지역 제어기들의 동작을 제어하기 위한 Global_Sync 메시지와 계획된 보행 궤적 데이터와 실험 결과 엔코더 데이터를 위한 Angle_Data, Encoder_Data 메시지와 센서 데이터를 위한 Sensor_Data 메시지가 정의되었다.

표2에서 정의된 ISHURO 메시지를 CAN을 이용하여 송수신하기 위해서는 CAN 데이터 형식으로 변환해 주어야 한다. 이 때 메시지 변환 과정에는 메시지의 우선순위를 지정하는 식별자(Identifier) 정의와 한 메시지 프레임의 크기 정보가 포함되어 설정된다. 표3은 지역 제어기 1의 CAN 메시지 식별자 및 데이터 크기를 나타낸다.

표 3. 지역 제어기의 CAN 데이터 메시지
Table 3. CAN messages of local controller

Message Name	ID	Direction	Data(byte)
Module_State	0x580	Send	1
	0x5c0	Receive	1
Output_I_State	0x6c0	Send	1
Walking_Position	0x600	Send	4
Jog_Move	0x680	Send	3
Global_Sync	0x4bf	Send	1
Angle_Data	0x500	Send	8
Encoder_Data	0x540	Receive	8
Sensor_Data	0x640	Receive	8

지역 제어기 1은 오른쪽 발목 요(ankle yaw) 관절부로 지역 제어기 중에서 우선순위가 가장 높고 반면 21번 지역 제어기가 우선순위가 가장 낮다. 메시지 별 우선순위가 가장 높은 메시지는 Global_Sync 메시지로 모든 지역 제어기를 동시에 제어하는 명령으로 CAN메시지에 포함된 데이터로 동작을 구분하게 된다. 다음의 우선순위는 Angle_Data이고 Output_I_State가 가장 낮다. 지역 제어기 2의 Module_State 메시지 ID는 0x581, 0x5c1 그리고 Output_I_State는 0x6c1를 가진다.

Module_State, Ouput_I_State, Global_Sync 메시지들의 크기는 1바이트이고 Jog_Move가 3바이트, Walking_Position이 4바이트로 구성되었다. Angle_Data, Encoder_Data와 Sensor_Data는 8바이트이고 8바이트는 CAN 메시지가 전송할 수 있는 최대 데이터의 크기이다. 수식(2)를 이용하여 메시지 별 전송 시간을 구하면 표 4와 같다.

표 4. 메시지 전송 시간
Table 4. Transmission time of messages

Message Name	Data(bits)	$C_i(\mu s)$	
		500kbps	1Mbps
Module_State	63	126	63
Output_I_State	63	126	63
Walking_Position	92	184	92
Jog_Move	82	164	82
Global_Sync	63	126	63
Angle_Data	130	260	130
Encoder_Data	130	260	130
Sensor_Data	130	260	130

8바이트 데이터를 전송하는 Angle_Data, Encoder_Data, Sensor_Data는 CAN 메시지의 길이가 130 비트로 길이가 가장 길고 따라서 버스의 전송 속도를 500kbps, 1Mbps에 전송하였을 때, 각각 260us 와 130us 가 걸리는 것을 확인할 수 있었다.

3.3 ISHURO의 동작 모드

ISHURO의 보행 궤적은 3초를 한 보(step)로 한다. 우선

모의실험기 에서 미리 경유점 정보를 입력하고 그 정보를 이용하여 5차 스플라인 곡선으로 로봇의 보행 궤적을 생성한다. 생성된 궤적에서 로봇의 역기구학 식을 이용하여 각 구동부의 각도 데이터를 구한다.

로봇 구동부 모터로는 A-max26, Re-max24 두 종류가 사용되었고, 각 모터의 입력 엔코더는 한 바퀴당 1000 pulse, 512 pulse이다. 구동부 모터의 입력 엔코더 값으로 변환된 각도 데이터는 주 제어기에서 지역 제어기로 보행 중에 CAN 버스를 통하여 전송 된다.

각도 데이터는 10ms 간격으로 계획되어 있고 실제 4바이트의 크기를 가지고 있다. CAN 메시지 정의에서 Angle_Data, Encoder_Data는 8바이트로 정의되었다. 따라서 하나의 CAN 메시지 프레임은 20ms의 데이터를 전송하게 된다. 즉, Angle_Data와 Encoder_Data의 전송 주기는 20ms가 된다. 로봇 보행 시 데이터 전송 주기20ms가 21개의 자유도를 가진 로봇의 CAN 버스의 데이터 이용 상태를 확인하는 기준이 된다.

로봇의 주 제어기와 지역 제어기간의 동작은 5가지 모드로 구성이 된다. 첫 번째 모드는 아이들(Idle)모드이다. 지역 제어기들은 파워 온이 되면 현재 위치를 유지하기 위한 제어를 시작하고 이 상태는 모터가 정 토크를 발생하는 상태이다. 두 번째 상태는 로봇의 다리부를 쭉 펴진 상태로 만들기 위해 각 지역 제어기가 출력 축 엔코더의 I-상을 찾는 모드로 I-상(Find-I-State) 모드이다. 세 번째 모드는 초기 위치 이동(Walking-Position) 모드이다. 출력 축 엔코더 I-상을 찾은 후 보행을 위한 자세를 취하기 위해 각 지역 제어기들은 주 제어기에서 전송된 출력 엔코더 펄스 값으로 지정된 위치로 정속도로 이동하게 된다. 네 번째 상태는 조그(Jog) 모드이다. 초기 위치로 이동한 로봇의 자세 교정을 위해 각 지역 제어기의 출력 축 엔코더 1펄스 단위로 움직일 수 있는 상태이다. 마지막 다섯 번째 보행(Walking)모드이다. Global_Sync 메시지를 수신한 모든 지역 제어기들이 수신되는 입력 엔코더 값으로 제어를 시작하는 상태라고 할 수 있다. 전체적으로 그림 5와 같이 CAN 메시지와 제어기의 모드 상태를 표현 할 수 있다.

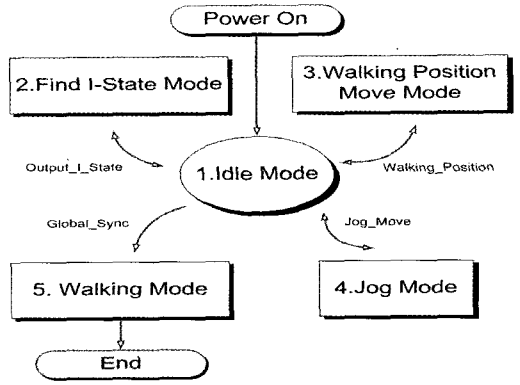

그림 5. 로봇제어기의 동작모드
Fig. 5. Operation mode of a robot

모든 제어기는 파워 온 상태에서 아이들 모드로 들어가게 되고 사용자 명령에 따라 모드 순서에 의해 보행 초기화 과정을 수행한다. 보행 초기화가 완료된 상태에서 사용자는 Global_Sync명령을 모든 지역 제어기로 전송하여 보행을 시작하게 한다. 계획된 보행이 완료되면 모든 동작을 마치게 되고 결과 데이터는 주 제어기에 저장되는 구조를 가지고 있다.

4. 모의 실험

우선 20ms 주기 동안 정의된 CAN 메시지들의 전송 상태와 지역 제어기의 여유 시간을 확인하게 위해 Matlab/Simulink를 이용하여 모의실험을 수행하였다. 그림 6은 모의실험기를 보여주고 있으며 개념을 단순화시키기 위해 세부 4부분으로 나누었다

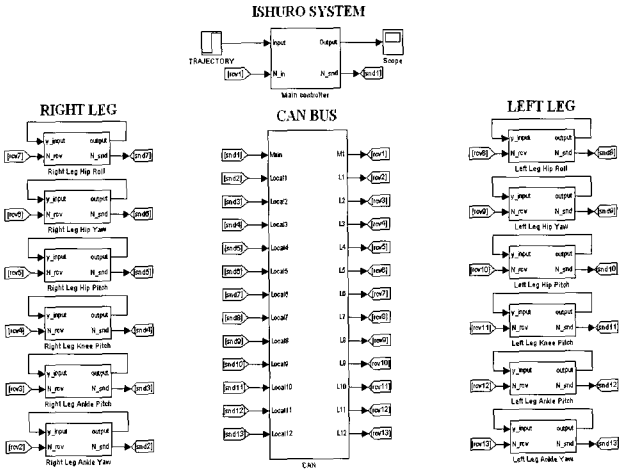

그림 6. ISHURO의 모의 실험기
Fig. 6. Simulator of ISHURO

구성된 모의 실험기는 1개의 주제어기와 12개의 지역 제어기를 가지고 있다. 즉 보행을 위한 좌우 다리 12개자유도만 고려하였다. 12 자유도에 대한 보행 궤적 전송 시 CAN 버스 상태를 모니터링 하여 버스 사용 상태를 확인하고 지역 제어기의 최대 여유 시간을 측정하였다. 모의 실험기는 실제 ISHURO 시스템과의 유사한 환경은 구성하기 위해 프로세서내의 연산 시간과 네트워크 송수신 시간을 확인할 수 있는 실시간 커널과 CAN 버스를 구성할 수 있는 네트워크 블록으로 구성되어 있다.

10ms로 샘플링 된 지역 제어기의 보행 궤적 데이터 2개로 하나의 CAN 데이터 프레임을 구성하였다. 따라서 주 제어기는 20ms 주기로 지역 제어기들로 데이터를 전송하게 되고 지역 제어기들은 데이터 수신 후 이전 주기에서 실행된 결과 데이터를 CAN 우선순위가 높은 지역 제어기1번(오른쪽 발목 요 관절)부터 순서대로 주 제어기로 결과 데이터를 전송하게 된다. 여기서 CAN 버스의 전송 속도는 1Mbps이고 1 frame의 크기는 130비트이다.

주 제어기에서 지역 제어기로 전송 되는 메시지의 우선순위가 지역 제어기에서 주 제어기로 보내지는 메시지보다 우선순위가 높기 때문에 보행 궤적 데이터의 전송이 끝난 후에 각 지역 제어기들은 CAN 메시지 우선순위에 의해 주 제어기로 결과 데이터를 전송하게 된다.

그림 7은 시간 계획된 데이터 전송 시 CAN 버스 상태를 나타낸다. 주 제어기에서 궤적 데이터를 전송하고 지역 제어기에서는 엔코더 데이터를 전송한다. 전체 12초 보행 중에서 처음부터 0.045s사이의 CAN 버스 상태를 나타내고 있다. 최대 데이터 전송 상태는 0.5의 크기로 나타내며 0.25의 크기는 CAN메시지의 우선순위에 의한 버스 중재 상태를 의미한다. 가장 아래 그래프가 주제어기에 의한 데이터 전송을 나타내고 위로 올라가면서 차례대로 지역 제어기1~12번순의 데이

터 전송 상태를 나타낸다. 20ms 주기로 주 제어기가 지역 제어기로 궤적 데이터 전송을 마치면 우선순위에 의해 각 지역 제어기의 결과 데이터 전송이 시작된다. 처음부터 0.02s사이에는 지역 제어기들은 데이터를 전송하지 않고 한 주기가 지난 후부터 주 제어기에서 보낸 궤적을 수신한 후에 엔코더 데이터를 전송하게 된다. 0.02s 이후에 버스 상태는 0.02에서부터 0.04s까지의 주기의 상태가 연속적으로 나타나고 있다.

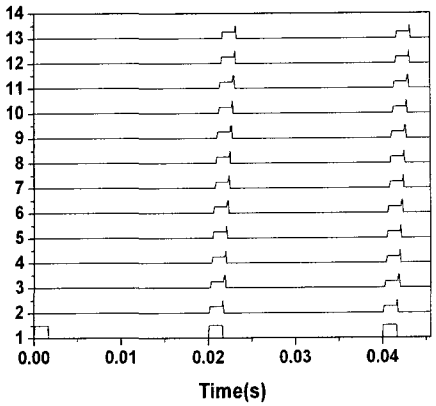

그림 7. 엔코더 및 궤적 전송 시 CAN 버스 상태
Fig. 7. CAN time sequence with encoders and trajectory data

한번 전송에서 CAN 버스 사용 시간은 3.274ms이고 각 지역 제어기들의 FSR센서 데이터 전송을 추가하면 4.896ms 이다. 그림8은 센서 데이터 전송을 포함한 CAN 상태를 나타낸다. 주 제어기에 의한 데이터 전송이 완료된 후 우선 순위에 따라 지역 제어기들의 엔코더 데이터 전송이 시작된다.

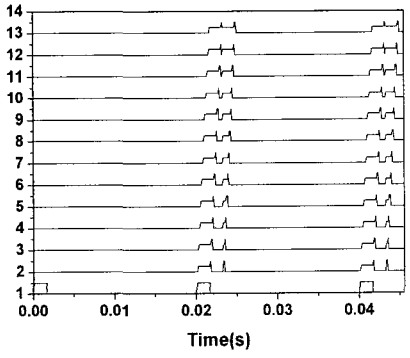

그림 8. 센서 데이터 전송시 CAN버스 상태
Fig. 8. CAN time sequence with sensor information

그리고 지역 제어기들은 센서 데이터를 전송한다. 첫 주기 0.0~0.02s동안은 주 제어기에 의한 궤적 데이터 전송만 수행된다. 0.02s이후 지역 제어기들은 궤적 데이터를 수신 후 엔코더 데이터를 전송하게 되고 그 다음 센서 데이터를 우선순위에 의해 전송하게 된다. 로봇은 총 21개의 자유도를 가지고 있으므로 위의 모의실험 결과를 적용하면 다리의 12 자유도와 상체 9 자유도에 대한 전송 속도를 1 Mbps로 사용한 경우 데이터 및 엔코더 결과 데이터 송수신에 8.19ms가 소요된다. 따라서 20ms주기 동안 지역 제어기 및 새로운 메시지 추가가 가능하여 이는 로봇이 어떠한 환경에 있을지라도 로봇의 통신에 의한 시스템 성능의 저하가 없음을 의미한다.

5. 동작 실험

실제 실험은 단위 걸음 당 3 초, 한 보폭 당 12 cm로 4걸음의 연속보행을 수행하였으며 그림9는 ISHURO의 다리부 12개의 자유도를 갖는 전체 외관을 보여준다. 로봇의 구동은 먼저 보행을 위한 초기 자세를 만들기 위해 로봇 외부 명령기에서 구동부의 출력 축 엔코더의 I-상을 검출하여 로봇이 똑바로 선 상태를 만들어 주고 보행을 위한 무릎을 구부린 자세를 취하기 위한 각 구동부의 초기 위치 값을 전송하여 보행 초기 상태를 완성한다.

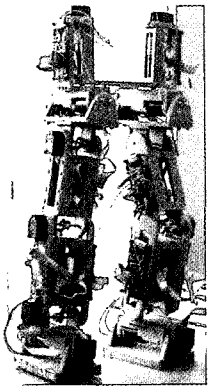

그림 9. ISHURO 로봇시스템
Fig. 9. ISHURO robot

초기화 과정을 마친 후 보행 시작을 알리는 사용자 명령에 의해 로봇은 보행을 수행한다. 보행 중에는 20ms주기로 데이터의 송수신이 이루어지고 보행을 마친 후 지역 제어기들의 엔코더 결과 데이터들은 주제어기에 저장된다. 보행 실험은 로봇의 보행 안정성을 고려하여 지지대에 장착하여 부하가 없는 상태에서 실험 되었다. 그림 10과 11은 각각 12초 보행한 후 요 관절과 피치 관절의 움직임을 나타낸 그림이다.

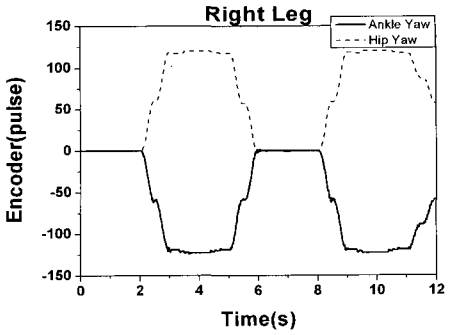

그림 10. 요 관절 움직임
Fig. 10. Yaw-direction movement of leg

여기서 메시지는 미리 메시지 스케쥴링을 한 후 보행 실험을 하였다. 전체 20ms 주기로 데이터를 수신하는 지역 제어기는 내부에 10ms 간격의 데이터를 가지고 1ms 단위의 주기로 나누어서 PID제어를 수행하였다. 반면 관절 움직임의 결과 값은 10ms주기로 샘플링 되어 2개의 데이터를 하나의 CAN메시지로 묶어 주 제어기로 전송하는 형태를 취하였다. 동작 결과 관절의 움직임이 실시간으로 불연속이 없이 부드럽게 표현되어 정해진 시간에 제어 데이터가 원활하게 전송되고 동작된 결과가 정확하게 전송됨을 확인할 수 있었다.

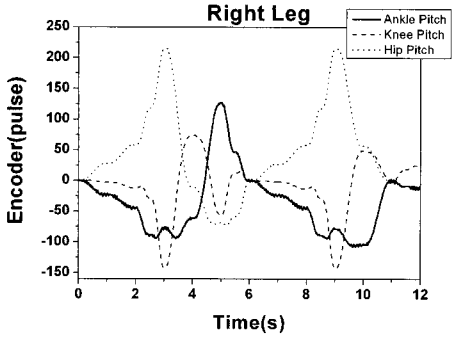

그림 11. 피치 관절 움직임
Fig. 11. Pitch-direction movement of leg

그림 12는 본 실험에 대한 ISHURO의 실제 보행 모습이고, 보행 실험은 3초 보행으로 4보 보행을 수행한 결과로써 각 사진은 1초 간격으로 찍은 사진을 나타낸다. 이때 PID 제어기 성능은 최대 오차가 0.0951rad, 오차의 평균은 0.0003rad, 오차의 분산은 0.054이었다.

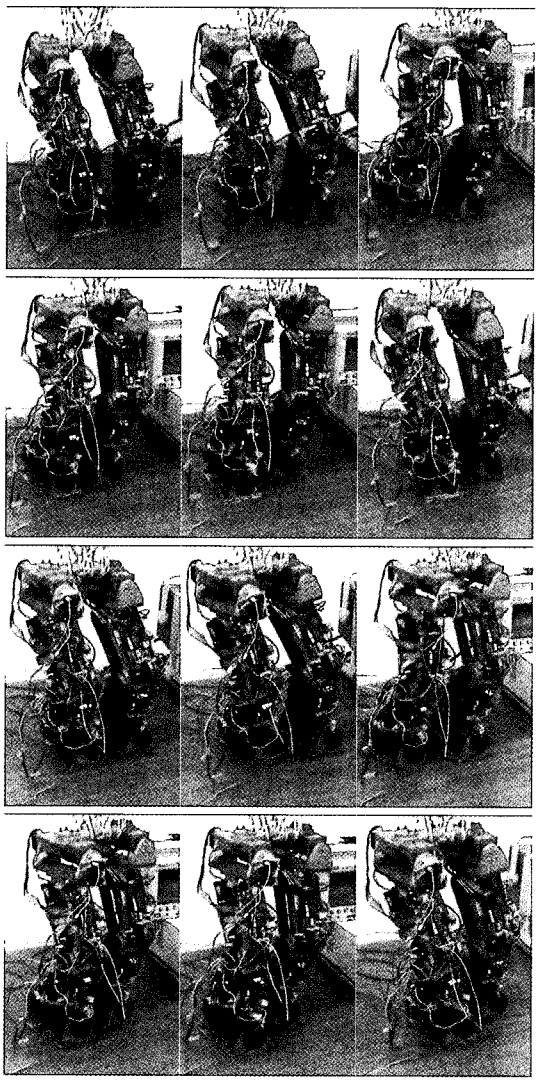

그림 12. ISHURO-I의 보행 실험
Fig. 12. Walking Experiment of ISHURO-I

6. 결 론

본 논문에서는 CAN 통신 프로토콜을 이용하여 휴머노이드 로봇의 지역 제어기들 간의 메시지를 신속, 정확하게 전달하여 실시간 통신을 가능하게 하는 네트워크 프로토콜에 대한 구현을 다루었다. 이러한 로봇은 기구학적으로 많은 자유도를 필요로 하고 안정적인 보행을 하기 위해서는 정밀한 모터 제어 및 제어기간의 빠른 데이터 전송 속도가 요구된다. 자체 제작된 휴머노이드 로봇 ISHURO는 내부에 주 제어기와 21개의 지역 제어기들이 CAN 버스로 연결된 분산 제어 구조를 가지고 있다. 주 제어기는 계획된 보행 궤적을 CAN 메시지화 하여 지역 제어기들로 전송하고 각 제어기들의 결과 데이터를 수신하여 저장하게 되는데 이때 메시지의 전송속도를 비롯 다양한 형태의 메시지 프로토콜에 대한 검증이 필요하다. 본 논문에서는 모의실험을 통해 계획된 메시지 스케줄링에 의한 데이터 송수신 시 CAN 버스 상태를 확인하여 실제 로봇의 보행 초기화 작업 및 보행 실험을 실시하였으며 주 제어기에 저장된 지역 제어기들의 결과 데이터와 지역 제어기에 저장된 데이터를 비교하여 통신 과정에서의 오류가 없음을 확인하였다.

향후 구동부 제어 안정성을 확보하기 위하여 다양한 종류의 센서가 장착될 시 보행 및 센서 데이터가 다수의 형태가 추가적으로 필요하며 이를 위한 다량의 정보를 실시간으로 처리할 수 있는 CAN 메시지 스케줄링 및 분산 처리 방법에 대한 연구가 필요하다.

참 고 문 헌

[1] N. Kanehira, T.U. Kawasaki, S. Ohta, T. Ismumi, T. Kawada, F. Kanehiro, S. Kajita, K. Kaneko, "Design and experiments of advanced leg module (HRP-2L) for humanoid robot (HRP-2) development," IEEE/RSJ International Conference on Intelligent Robots and System, vol.3, pp.2455-2460, Sept.30-Oct.5, 2002.

[2] T. Ishida, Y. Kuroki, "Motion and real-world sensing system of a small biped entertainment robot," Intelligent Control and Automation, WCICA 2004, vol.6, pp. 4834 - 4839, 2004.

[3] T. Fukaushima, Y. Kuroki, T. Ishida, "Development of a new actuator for a small biped walking entertainment robot," Power Electronics, Machines and Drives, PEMD 2004, vol.1, pp. 126 - 131, 2004.

[4] 조영조, 오상록, "지능형 서비스 로봇과 URC (Ubiquitous Robotic Companion)," 한국통신학회지, vol.21, pp.13-21, Oct. 2004.

[5] Y. Kuroki, B. Blank, T. Mikami, P. Mayeux, A. Miyamoto, "Motion creating system for a small biped entertainment robot," Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol.2, pp.1394-1399, Oct.27-31, 2003.

[6] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, K. Fujimura, "The intelligent ASIMO: system overview and in-

tegration," IEEE/RSJ International Conference on Intelligent Robots and Systems, vol.3, pp.2478-2483, Sept.30-Oct.5, 2002.
[7] M. Santos, M. Stemmer, F. Vasques, "Schedulability analysis of messages in a CAN network applied to an unmanned airship," IEEE 2002 28th Annual Conference of the IECON 02, vol.3, pp.1909-1914, Nov.5-8, 2002.
[8] J.A. Fonseca, L.M. Almeida, "Using a planning scheduler in the CAN network," Proceedings of the 1999 IEEE International Conference on Emerging Technologies and Factory Automation, vol.2, pp.815-821, Oct.18-21, 1999.
[9] 박진우 외 4인, "CAN을 이용한 분산 제어 구조를 가지는 이동 로봇 구현," 1999년도 대한전자공학회 한국통신학회 부산경남지부 합동 학술논문발표회 논문집, 251-255, 1999.
[10] 최영진, 류제훈, 오용환, 유범재, 오상록, "소형 휴머노이드 '베이비봇' 개발," ICASE Magazine, Vol. 10, No. 4, pp. 29-34, July, 2004.
[11] Jung-Yup Kim, Ill-Woo Park, Jungho Lee, Min-Su Kim, Baek-kyu Cho, and Jun-Ho Oh, "System Design and Dynamic Walking of Humanoid Robot KHR-2," 2005 IEEE Int. Conference on Robotics and Automation, Spain, pp. April, 1443-1448, 2005.
[12] Ni Xiaodong, Zhang Yanjun, "Determining message delivery delay of controller area networks," Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering, vol.2, pp. 767 - 771, Oct.28-31, 2002.
[13] L. Sha, R. Rajkumar, J.P. Lehoczky, "Priority inheritance protocols: an approach to real-time synchronization," IEEE Transactions on Computers, vol.39, pp. 1175 - 1185, Sept. 1990.
[14] S, Misbahuddin, N. Al-Holou, "Efficient data communication techniques for controller area network (CAN) protocol," Computer Systems and Applications, 2003. Book of Abstracts. ACS/IEEE International Conference on, pp.22, 14-18 July 2003.
[15] 이경창, 김정희, 이홍희, "스마트 홈을 위한 CAN 기반 화재 감지 시스템의 구현," 제어 • 자동화 • 시스템공학 논문지, 제10권, 제8호, Aug. 2004.
[16] K.W. Tindell, H. Hansson, A.J. Wellings, "Analysing real-time communications: controller area network (CAN)," Proceedings of Real-Time Systems Symposium, pp.259-263, Dec.7-9, 1994.
[17] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, K.E. Arzen, "How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime," Control Systems Magazine, IEEE, pp. 16-30, vol.23, Issue:3, June 2003.

저 자 소 개

이보희(Lee Bo-Hee)
1996년 : 인하대학교 기계공학과 자동화 전공 박사 졸업.
1997년~현재 : 세명대학교 전기공학과 부교수.

관심분야 : 휴머노이드 로봇, 인공지능, 제어 시스템
Phone : 043-643-1305
Fax : 043-644-6966
E-mail : bhlee@semyung.ac.kr

공정식(Kong Jung-Shik)
1998년 : 인하대학교 자동화공학과 학사 졸업.
2006년 : 인하대학교 자동화공학과 박사 졸업

관심분야 : 유전 알고리즘, 휴머노이드 로봇, 모터 제어
Phone : 032-860-8923
Fax : 032-863-5822
E-mail : tempus@dreamwiz.com

김진걸 (Kim Jin-Geol)
1988년 : Univ. of Iowa 전기 및 컴퓨터 공학과 박사 졸업.
1988년~현재 : 인하대학교 전자전기공학부 정교수

관심분야 : 로봇 제어, 컴퓨터 및 비선형 제어
Phone : 032-860-7384
Fax : 032-863-5822
E-mail : john@inha.ac.kr